Technical Report: Hate Speech Detection Using GPT-2

Amir Fasil

May 7, 2025

1 Introduction

This technical report documents the development and evaluation of a hate speech detection system using GPT-2, comparing base model performance with few-shot prompting against a fine-tuned version using LoRa. The project explores the effectiveness of different prompting techniques and fine-tuning approaches for content moderation tasks.

2 Task and Dataset Documentation

2.1 Task Definition

The primary task involves binary classification of text samples into:

- Hate speech (1)
- Non-hate speech (0)

2.2 Dataset Specification

- **Source**: waalbannyantudre/hate-speech-detection-curated-dataset
- **Size**: 2000
- **Class Distribution**: 50

- **Preprocessing**: load$_{p}repfunctionfordatacleaning$

Dataset(features: ['Content', 'labels', '$_{index_{l}evel_{0_{'],num_{r}ows:2000}}}$)

3 Methodology

3.1 Model Selection

- Initially selected Llamma 2 because of its superior capability however since the VRAM amount i have in google colab is 15GB and the base model alone uses 14GB it doesn't have enough resourse for finetuning
- Llamma 2 7 Billiion parameters while GPT 2 137M parameters
- Rationale: For the reason mentioned above I choose GPT 2 large

3.2 Experimental Setup

3.2.1 Approach 1: Base Model with Few-Shot Prompting

- Used unmodified GPT-2 base model
- Implemented few-shot prompting with evaluation dataset
- Prompt template: f"""

  Example 1: Text: "I hate all people from country X." Label: Hate Speech

  Example 2: Text: "I love everyone, no matter where they are from." Label: Non-Hate Speech

  Example 3: Text: "People who support this political party are all idiots." Label: Hate Speech

  Example 4: Text: "Everyone has the right to be respected." Label: Non-Hate Speech

  Now, classify the following text: Text: text Label:

  """

3.2.2 Approach 2: Fine-Tuned Model with Zero-Shot Prompting

- Fine-tuning method: Supervised fine-tuning with LoRa
- Attempted QLoRa but faced significant dependency issues
- Training parameters:
  - Learning rate: 2e-5
  - Batch size: 4
  - Epochs: [3
  - LoRa parameters (rank, alpha): (8,16)
- Hardware: GPU T4 with 15GB VRAM

4 Prompt Engineering Approaches

4.1 Few-Shot Prompting (Base Model)

- Advantages: No training required, flexible to task changes
- Limitations: some of the parameters are not initialized

4.2 Zero-Shot Prompting (Fine-Tuned Model)

- Structure: f"Classify this text strictly as either 'hate$_{s}peech' or' not_{h}ate_{s}peech'$: $text$" $Advantages : Leveragesmodel'slearnedrepresentations$

- Limitations: some of the parameters are not initialized

5 Comparative Analysis

Table 1: Performance Comparison

Metric	Few-Shot (Base)	Zero-Shot (Fine-Tuned)
Accuracy	50%	60%
evaluation loss	0.9198812246322632	0.6752246022224426

Key observations:

- Fine-tuned model showed 10% absolute improvement in accuracy
- I know It will do better if we used Llamma 2

6 Theoretical Discussion

6.1 Few-Shot vs. Zero-Shot Performance

- Base model limitations explain 50% accuracy (near random for binary classification)
- Fine-tuning enables better zero-shot performance by adapting model parameters

6.2 LoRa vs. QLoRa Considerations

- Successful LoRa implementation despite QLoRa challenges
- Quantization benefits were missed due to QLoRa dependency issues
- Memory efficiency trade-offs between approaches

6.3 Model Selection Implications

- GPT-2's limitations compared to LLaMA-2
- Potential performance gains with more modern architectures

7 Conclusion and Future Work

7.1 Key Findings

- Fine-tuning with LoRa provided measurable improvements over few-shot baseline
- Prompt engineering approach should match model capabilities
- Infrastructure challenges impacted technique selection

7.2 Future Directions

- Migrate to LLaMA-2 or more capable base model
- Resolve QLoRa dependency issues for memory efficiency
- Explore hybrid prompting approaches
- Expand dataset and augment training samples